

Bulk Data Transfer Techniques for High-Speed Wide-Area Networks

Brian L. Tierney

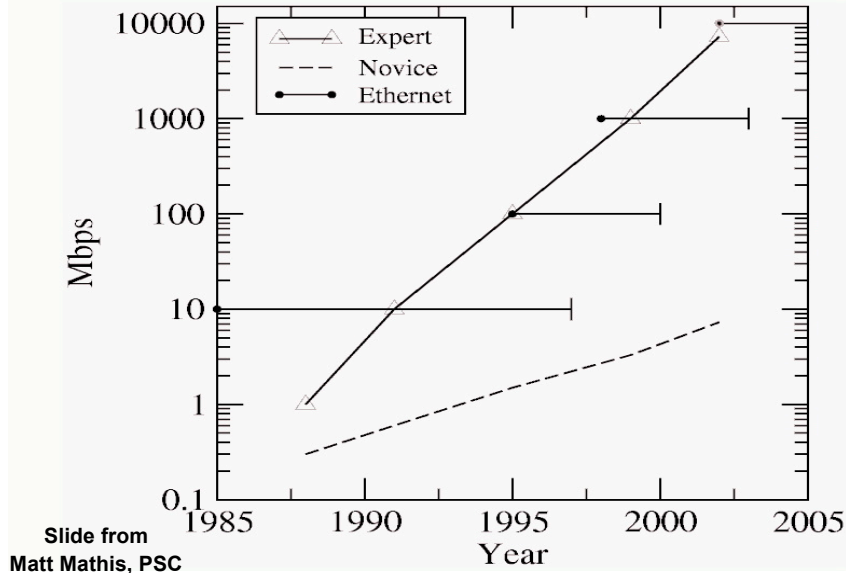
Distributed Systems Department
Lawrence Berkeley National Laboratory

**Why
does the
Network
seem so
slow?**





Wizard Gap



Today's Talk



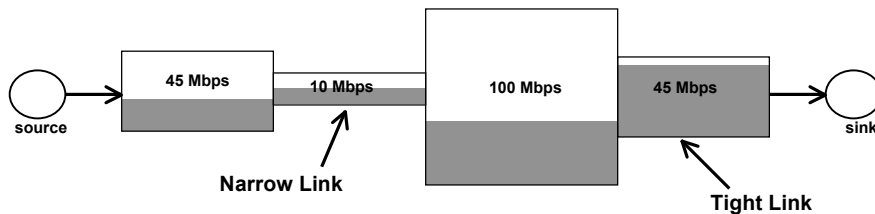
- **This talk will cover:**
 - Some Information to help you become a “wizard”
 - Work being done so you don’t have to be a wizard
- **Outline**
 - TCP Tuning Techniques (focus is on Linux)
 - TCP Issues
 - Bulk Data Transfer Tools
 - Network Monitoring Tools

Time to Copy 1 Terabyte

- 10 Mbps network : 300 hrs (12.5 days)
- 100 Mbps network : 30 hrs
- 1 Gbps network : 3 hrs
- 10 Gbps network : 20 minutes
 - need very fast disk array for this
- Compare these speeds to:
 - USB 2.0 portable disk
 - + 60 MB/sec (480 Mbps) peak
 - + 20 MB/sec (160 Mbps) reported on line
 - + 5-10 MB/sec reported by colleagues
 - + 15-40 hours to load 1 TeraByte

Terminology

- The term “Network Throughput” is vague and should be avoided
 - **Capacity: link speed**
 - + Narrow Link: link with the lowest capacity along a path
 - + Capacity of the end-to-end path = capacity of the narrow link
 - **Utilized bandwidth: current traffic load**
 - **Available bandwidth: capacity – utilized bandwidth**
 - + Tight Link: link with the least available bandwidth in a path
 - **Achievable bandwidth: includes protocol and host issues**

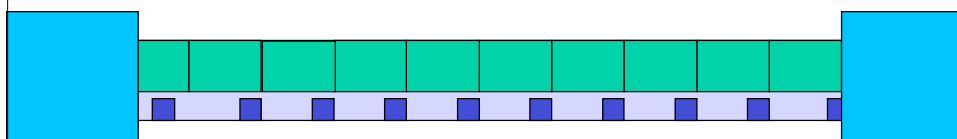


More Terminology

- **Latency:** time to send 1 packet from the source to the destination
- **RTT:** Round-trip time
- **Bandwidth*Delay Product = BDP**
 - The number of bytes in flight to fill the entire path
 - Example: 100 Mbps path; ping shows a 75 ms RTT
+ $BDP = 100 * 0.075 / 2 = 3.75 \text{ Mbits (470 KB)}$
- **LFN:** Long Fat Networks
 - A network with a large BDP

TCP Window and ACKs

Small TCP Buffer Size



Data Packet

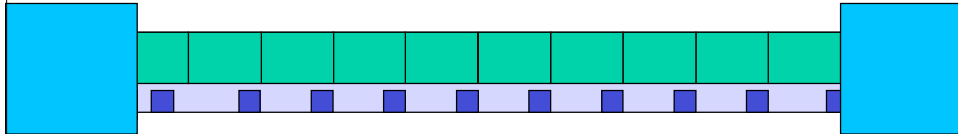


Acknowledgement

Animated Slide from Globus Project, ANL

TCP Window and ACKs

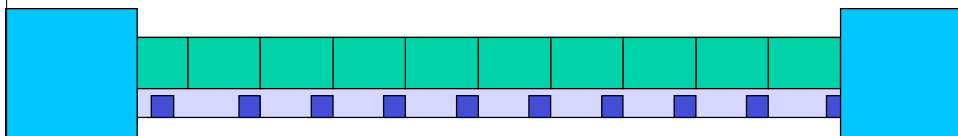
Half Full (1 trip)



Animated Slide from Globus Project, ANL

TCP Window and ACKs

Optimized TCP Buffer Size



Animated Slide from Globus Project, ANL

TCP Performance Issues

- Getting good TCP performance over high-latency high-bandwidth networks is not easy!
- You must keep the TCP window full, and the size of the window is directly related to the network latency
 - Example: from LBNL to BNL the narrow link is 1000 Mbits/sec, and the one-way latency is 40 ms
 - Need $(1000 / 8) * .04 \text{ sec} = 5 \text{ MBytes}$ of data “in flight” to fill the TCP window
- Easy to compute max throughput:
 - Throughput = buffer size / latency
 - eg: 64KB buffer / 40 ms path = 1.6 Kbytes (12.8 Kbits) / sec

Sample Results: LBL to BNL

- Using the right tool is very important
 - **scp / sftp : 10 Mbps**
 - + standard Unix file copy tools
 - + fixed 64 KB TCP window in openssl
 - **ftp : 400-500 Mbps**
 - + using Linux > 2.6.12 or Windows Vista
 - **parallel GridFTP: 800-900 Mbps**

TCP Buffer Size

- The TCP Buffer must be large enough to accommodate a full TCP window for the path
- 2 Issues:
 - **Maximum System Buffers are too small**
 - + 128 KB to 1MB, depending on OS
 - + Increasing this is quite easy, but requires superuser access
 - **Application Buffers even smaller**
 - + Linux 2.6 and Window Vista do “autotuning”
 - + All other OS's , applications must set this explicitly

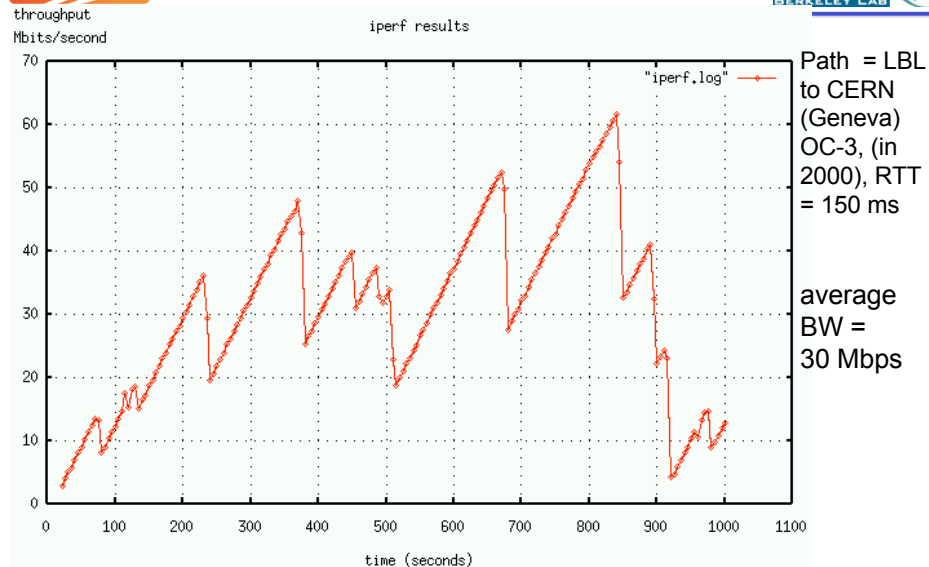
Buffer Size Example

- **Optimal Buffer size formula:**
 - $\text{buffer size} = \text{bandwidth} * \text{RTT}$
- ping time (RTT) = 50 ms
- **Narrow link = 500 Mbps (62 Mbytes/sec)**
 - e.g.: the end-to-end network consists of all 1000 BT ethernet and OC-12 (622 Mbps)
- **TCP buffers should be:**
 - $.05 \text{ sec} * 62 = 3.1 \text{ Mbytes}$

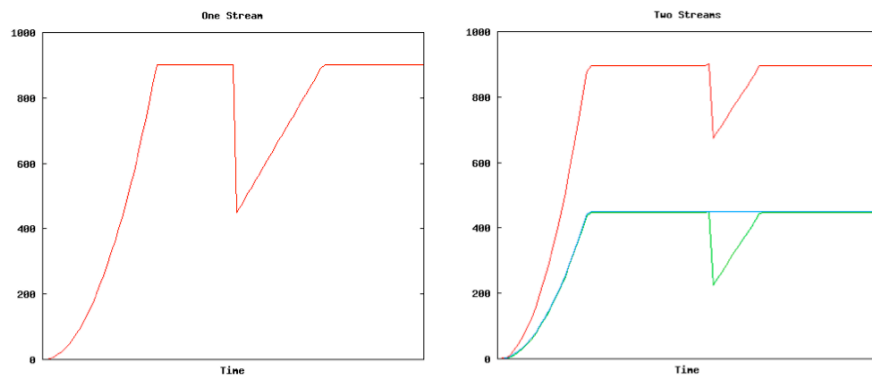
Sample Buffer Sizes

- LBL to
 - SLAC (RTT = 2 ms, narrow link = 1000 Mbps) : 256 KB
 - BNL: (RTT = 80 ms, narrow link = 1000 Mbps): 10 MB
 - CERN: (RTT = 165 ms, narrow link = 1000 Mbps): 20.6 MB
- Note: default buffer size is usually only 64 KB, and default maximum buffer size for is only 256KB
 - Linux Autotuning default max = 128 KB (recently increased to 1 MB);
- 10-150 times too small!
- Home DSL, US to Europe (RTT = 150, narrow link = 2 Mbps): 38 KB
 - Default buffers are OK.

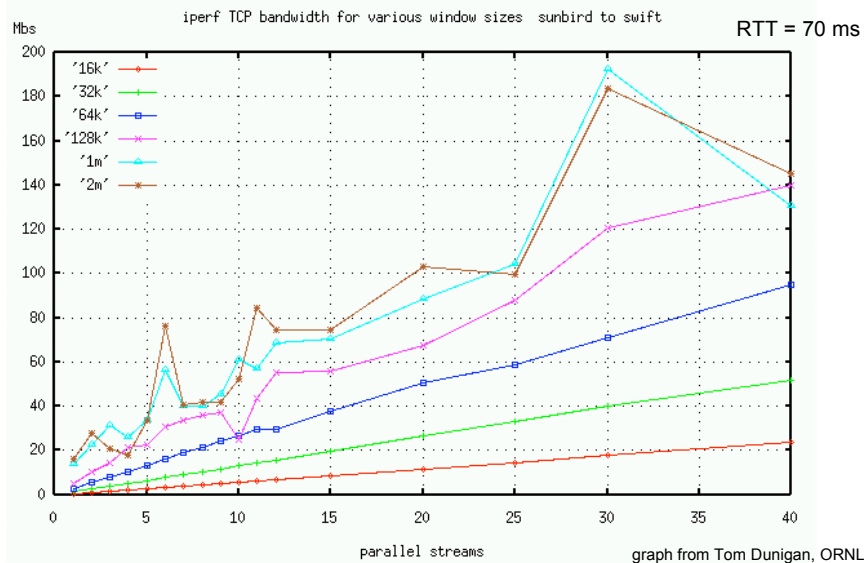
More Problems: TCP congestion control



Parallel Streams Can Help

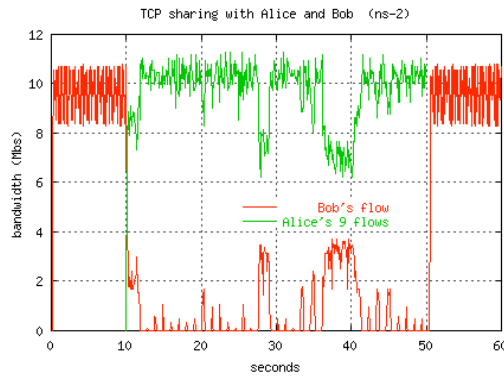


Parallel Streams Results with various TCP window sizes



Parallel Streams Issues

- Potentially unfair
- Places more load on the end hosts
- But they are necessary when you don't have root access, and can't convince the sysadmin to increase the max TCP buffers

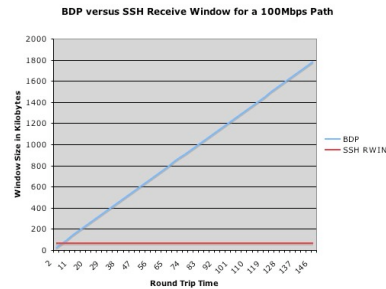


graph from Tom Dunigan, ORNL

Bulk Transfer Tools

scp

- Don't use scp to copy large files across a WAN!
 - scp has its own internal 64KB buffering/windowing that prevents it from ever being able to fill LFNs!
- Explanation of problem and openssh patch solution from PSC:
 - <http://www.psc.edu/networking/projects/hpn-ssh/>



GridFTP

- GridFTP from ANL has everything needed to fill the network pipe
 - Buffer Tuning
 - Parallel Streams
- Supports multiple authentication options
 - anonymous
 - ssh (available in current Globus Toolkit “development” release version 4.1.2)
 - X509
- Ability to define a range of data ports
 - helpful to get through firewalls
- Sample Use:
 - `globus-url-copy -p 4 sshftp://data.lbl.gov/home/mydata/myfile file://home/mydir/myfile`
- Available from:
 - <http://www.globus.org/toolkit/downloads/#development>

- **bbftp** (from the HEP “Babar”) project also everything needed to fill the network pipe
 - Buffer Tuning
 - Parallel Streams
- Supports ssh authentication options
- Supports on-the-fly compression
- Sample Use:
 - `bbftp -p 4`
`bbftp://data.lbl.gov/home/mydata/myfile`
`file://home/mydir/myfile`
- Available from: <http://doc.in2p3.fr/bbftp/>

- **Open Source:**
 - <http://filezilla.sourceforge.net/>
 - includes client and server
- **Features:**
 - ability to transfer multiple files in parallel
- **Issues:**
 - uses libopenssl in secure mode, so has buffer limitations

Special Purpose Data Transfer Tools

- **HPSS Tools: HSI and pftp**
 - both support buffer tuning and parallel transfers
 - + per destination buffer tuning must be done by HPSS admin
- **SRM from SDSC**
 - supports buffer tuning and parallel transfers

Other Issues

- **Firewalls**
 - many firewalls can't handle 1 Gbps flows
 - + designed for large number of low bandwidth flow
 - + some firewalls even strip out TCP options that allow for TCP buffers > 64 KB
- **Disk Performance**
 - In general need a RAID array to get more than around 500 Mbps
- **Other subtle issues start to come up at speeds above 2 Gbps**
 - Router buffering, TCP congestion control, disk tuning, etc.

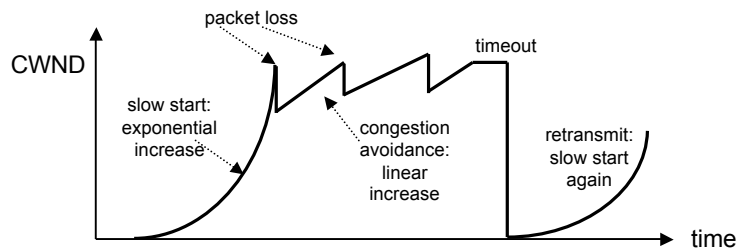
More Information

- <http://dsd.lbl.gov/WAN-bulk-transfer/>
- <http://dsd.lbl.gov/TCP-tuning/>
- email: BLTierney@LBL.GOV

Extra (advanced) slides

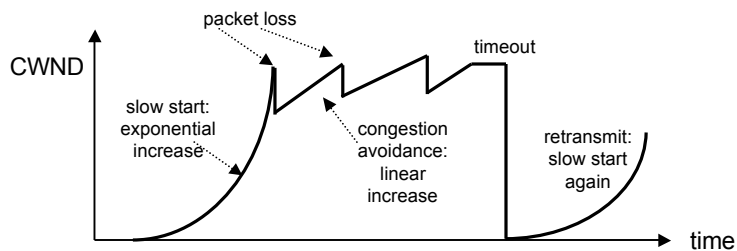
How TCP works: A very short overview

- **Congestion window (CWND) = the number of packets the sender is allowed to send**
 - The larger the window size, the higher the throughput
 - + $\text{Throughput} = \text{Window size} / \text{Round-trip Time}$
- **TCP Slow start**
 - exponentially increase the congestion window size until a packet is lost
 - + this gets a rough estimate of the optimal congestion window size



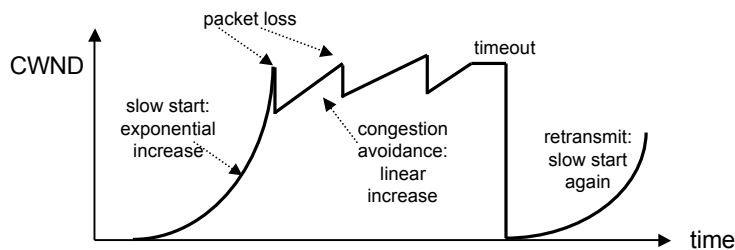
TCP Overview

- **Congestion avoidance**
 - additive increase: starting from the rough estimate, linearly increase the congestion window size to probe for additional available bandwidth
 - multiplicative decrease: cut congestion window size aggressively if a timeout occurs



TCP Overview

- **Fast Retransmit:** retransmit after 3 duplicate acks (got 3 additional packets without getting the one you are waiting for)
 - this prevents expensive timeouts
 - no need to go into “slow start” again
- **At steady state, CWND oscillates around the optimal window size**
- **With a retransmission timeout, slow start is triggered again**



Setting the TCP buffer sizes

- **It is critical to use the optimal TCP send and receive socket buffer sizes for the link you are using.**
 - Recommended size = $2 \times \text{Bandwidth Delay Product (BDP)}$
 - if too small, the TCP window will never fully open up
 - if too large, the sender can overrun the receiver, and the TCP window will shut down
- **Default TCP buffer sizes are way too small for this type of network**
 - default TCP send/receive buffers are typically 64 KB
 - tuned buffer for LBL to BNL link: 10 MB
 - + 150X bigger than the default buffer size
 - with default TCP buffers, you can only get a small % of the available bandwidth!

- Need to adjust system max TCP buffer
 - Example: for Linux add the entries below to the file `/etc/sysctl.conf`, and then run `"sysctl -p"`
- ```
increase TCP max buffer size
net.core.rmem_max = 16777216
net.core.wmem_max = 16777216
increase Linux autotuning TCP buffer limits
min, default, and max number of bytes to use
net.ipv4.tcp_rmem = 4096 87380 16777216
net.ipv4.tcp_wmem = 4096 65536 16777216
```
- Similar changes needed for other Unix OS's
  - For more info, see: <http://dsd.lbl.gov/TCP-Tuning/>

- The optimal buffer size is twice the bandwidth\*delay product of the link:  
$$\text{buffer size} = 2 * \text{bandwidth} * \text{delay}$$
- The ping program can be used to get the delay
  - e.g.: `>ping -s 1500 lxplus.cern.ch`  
1500 bytes from lxplus012.cern.ch: icmp\_seq=0. time=175. ms  
1500 bytes from lxplus012.cern.ch: icmp\_seq=1. time=176. ms  
1500 bytes from lxplus012.cern.ch: icmp\_seq=2. time=175. ms
- *pipechar* or *pathrate* can be used to get the bandwidth of the slowest hop in your path. (see next slides)
- Since ping gives the round trip time (RTT), this formula can be used instead of the previous one:  
$$\text{buffer size} = \text{bandwidth} * \text{RTT}$$

## Network Monitoring Tools

### traceroute

```
>traceroute pcgiga.cern.ch
traceroute to pcgiga.cern.ch (192.91.245.29), 30 hops max, 40 byte packets
 1 iri00gw-r2.lbl.gov (131.243.2.1) 0.49 ms 0.26 ms 0.23 ms
 2 eri00gw.lbl.gov (131.243.128.5) 0.68 ms 0.54 ms 0.54 ms
 3 198.129.224.5 (198.129.224.5) 1.00 ms *d9* 1.29 ms
 4 lbl2-ge-lbnl.es.net (198.129.224.2) 0.47 ms 0.59 ms 0.53 ms
 5 snv-lbl-oc48.es.net (134.55.209.5) 57.88 ms 56.62 ms 61.33 ms
 6 chi-s-snv.es.net (134.55.205.102) 50.57 ms 49.96 ms 49.84 ms
 7 arl-chicago-esnet.cern.ch (198.124.216.73) 50.74 ms 51.15 ms 50.96 ms
 8 cernh9-pos100.cern.ch (192.65.184.34) 175.63 ms 176.05 ms 176.05 ms
 9 cernh4.cern.ch (192.65.185.4) 175.92 ms 175.72 ms 176.09 ms
10 pcgiga.cern.ch (192.91.245.29) 175.58 ms 175.44 ms 175.96 ms
```

Can often learn about the network from the router names:

**ge** = Gigabit Ethernet

**oc48** = 2.4 Gbps (**oc3** = 155 Mbps, **oc12**=622 Mbps)

- **iperf** : nice tool for measuring end-to-end TCP/UDP performance
  - <http://dast.nlanr.net/Projects/Iperf/>
  - Can be quite intrusive to the network
- **Example:**
  - **Server:** `iperf -s -w 2M`
  - **Client:** `iperf -c hostname -i 2 -t 20 -l 128K -w 2M`

```
Client connecting to hostname
[ID] Interval Transfer Bandwidth
[3] 0.0- 2.0 sec 66.0 MBytes 275 Mbits/sec
[3] 2.0- 4.0 sec 107 MBytes 451 Mbits/sec
[3] 4.0- 6.0 sec 106 MBytes 446 Mbits/sec
[3] 6.0- 8.0 sec 107 MBytes 443 Mbits/sec
[3] 8.0-10.0 sec 106 MBytes 447 Mbits/sec
[3] 10.0-12.0 sec 106 MBytes 446 Mbits/sec
[3] 12.0-14.0 sec 107 MBytes 450 Mbits/sec
[3] 14.0-16.0 sec 106 MBytes 445 Mbits/sec
[3] 16.0-24.3 sec 58.8 MBytes 59.1 Mbits/sec
[3] 0.0-24.6 sec 871 MBytes 297 Mbits/sec
```

- **Nice tools from Georgia Tech:**
  - **pathrate:** measures the capacity of the narrow link
  - **pathload:** measures the available bandwidth
- **Both work pretty well.**
  - pathrate can take a long time (up to 20 minutes)
  - These tools attempt to be non-intrusive
- **Open Source; available from:**
  - <http://www.pathrate.org/>

## Duplex Mismatch Issues

- A common source of trouble with Ethernet networks is that the host is set to full duplex, but the Ethernet switch is set to half-duplex, or visa versa.
- Most newer hardware will auto-negotiate this, but with some older hardware, auto-negotiation sometimes fails
  - result is a working but very slow network (typically only 1-2 Mbps)
  - best for both to be in full duplex if possible, but some older 100BT equipment only supports half-duplex
- NDT is a good tool for finding duplex issues:
  - <http://e2epi.internet2.edu/ndt/>

## Recent TCP Work

## Proposed TCP Modifications

- Well known fact that TCP does not scale to high-speed networks
- Many Proposed Solutions:
  - High Speed TCP: Sally Floyd  
+ <http://www.icir.org/floyd/hstcp.html>
  - BIC/CUBIC:  
+ <http://www.csc.ncsu.edu/faculty/rhee/export/bitcp/>
  - LTCP (Layered TCP)  
+ <http://students.cs.tamu.edu/sumitha/research.html>
  - HTCP: (Hamilton TCP)  
+ <http://www.hamilton.ie/net/htcp/>
  - Scalable TCP  
+ <http://www-lce.eng.cam.ac.uk/~ctk21/scalable/>

## Linux 2.6.12 Results

- BIC-TCP is included in Linux 2.6
  - un-tuned results up to 100x faster!

| Path                       | Linux 2.4<br>Un-tuned | Linux 2.4<br>Hand-tuned | Linux 2.6<br>with BIC | Linux 2.6,<br>no BIC |
|----------------------------|-----------------------|-------------------------|-----------------------|----------------------|
| LBL to ORNL<br>RTT = 67 ms | 10 Mbps               | 300 Mbps                | 700 Mbps              | 500 Mbps             |
| LBL to PSC<br>RTT = 83 ms  | 8 Mbps                | 300 Mbps                | 830 Mbps              | 625<br>Mbps          |
| LBL to IE<br>RTT = 153 ms  | 4 Mbps                | 70 Mbps                 | 560 Mbps              | 140 Mbps             |

- Results = Peak Speed during 3 minute test
- Must increase default max TCP send/receive buffers
- Sending host = 2.8 GHz Intel Xeon with Intel e1000 NIC

## Linux 2.6.12 Results

